# IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Date: April 13, 2004
Express Mail: ER837823832US

In re application of:  **C. Conan, et al**

Serial No.:  **09/772,474**

Filed:  **January 30, 2001**

Docket No.:  **CA919990016-US1**

Board of Patent Appeals and Interferences
Alexandria, VA 22213-1450
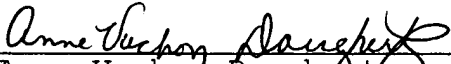
## TRANSMITTAL OF APPEAL BRIEF UNDER 37 CFR 1.192

Transmitted herewith, in triplicate is an Appeal Brief with respect to the Notice of Appeal filed February 13, 2004 for the above-identified patent application.

This Appeal Brief is being filed on behalf of other than a small entity.

Authorization is given to charge amount of $330.00, for filing a Brief in support of appeal in accordance with 37 CFR 1.17(f), to Deposit Account 50-0510.  A duplicate copy of this authorization is enclosed.

The Commissioner is hereby authorized to charge any required additional fee, and charge back any overpayment, to Deposit Account No. 50-0510.

Respectfully submitted,
C. Conan, et al

Anne Vachon Dougherty
Registration No. 30,374
Tel. (914) 962-5910

# IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

| | | |
|---|---|---|
| In Re Application of | : | April 13, 2004 |
| C. Conan, et al | : | Group Art No.: 2863 |
| Serial No. 09/772,474 | : | Examiner: T. S. Lau |
| Filed: January 30, 2001 | : | for IBM Corporation |
| | | Anne Vachon Dougherty |
| Title: AUTOMATED TESTING OF | | 3173 Cedar Road |
| COMPUTER SYSTEM COMPONENTS | | Yorktown Heights, NY 10598 |

Board of Patent Appeals and Interferences
Alexandria, VA 22313-1450

## APPEAL BRIEF (37 CFR 1.192)

Appellants hereby appeal to the Board of Patent Appeals and Interferences from the decision dated December 15, 2003 of the Primary Examiner finally rejecting Claims 1, 2, and 5-24 in the above application, and respectfully request that the Board of Patent Appeals and Interferences consider the

arguments presented herein and reverse the Examiner's rejections.

## I. REAL PARTY IN INTEREST

The appeal is made on behalf of Applicants who are real parties in interest with respect to the subject patent application.

## II. RELATED APPEALS AND INTERFERENCES

There are no pending related appeals or interferences with respect to the subject patent application.

## III. STATUS OF CLAIMS

There are twenty-two (22) claims pending in the subject patent application, numbered 1, 2, and 5-24. No claims stand allowed. All of Claims 1, 2, and 5-24 stand rejected.

A complete copy of the claims involved in the appeal is attached hereto.

# IV. STATUS OF AMENDMENTS

The status of the prosecution of the application is as follows:

| | | |
|---|---|---|
| March 2, 2001 | - | Preliminary Amendment, canceling Claims 3-4 |
| October 23, 2002 | - | Office Action, rejecting all pending claims |
| January 23, 2003 | - | Amendment, amending all independent claims |
| April 1, 2003 | - | Final Office Action, citing new grounds for rejecting Claims 1, 2, 5-13, and 15-24 and objecting to Claim 14 |
| July 1, 2003 | - | RCE and Amendment filed, amending all independent Claims |
| August 6, 2003 | - | Office Action, rejecting all pending claims, citing new grounds for rejection |
| November 6, 2003 | - | Amendment, amending all independent claims |
| December 15, 2003 | - | Final Office Action, rejecting all pending claims |
| February 13, 2004 | - | Notice of Appeal filed |

## V. SUMMARY OF INVENTION

The subject invention is a method, system, and program storage device for automated testing of software having the following:

(i) a test bucket for storing sets of test data,

(ii) a job receiver process, for accepting test requests from a user, each test request comprising an identifier for selecting test data from the test bucket,

(iii) a resource process and resource pool for managing system resource data to indicate resources available for software testing on a set of client computer systems, and

(iv) a job execution process for creating test execution script data based on the test data identified in the test request and on the available resources, wherein the job execution process receives the test request from the job receiver process, dynamically creates the test execution script based upon the resource pool indicating the availability of resources required for the execution of the test on one or more of the set of client computer systems, and initiates

testing by forwarding the test execution script data to the appropriate one or more of the set of client computer systems, and wherein the system server component has means for accepting and storing test results from the set of client computer systems.

As is expressly recited in all of the pending claims, the job execution process dynamically creates test execution script data based on the test data identified in the test request and based on the pool of resources which are dynamically determined to be available for executing the test on one or more client computers.

## VI. STATEMENT OF ISSUES OF APPEAL

There following issues are on appeal:

(1) whether the Examiner erred in applying the teachings of the Brouwer patent to the claim language;

(2) whether the Examiner erred in concluding that Claims 1-2, 5-13, and 15-24 are unpatentable over the Brouwer patent in view of the teachings of the Haley patent; and

(3) whether the Examiner erred in concluding that Claim 14 is unpatentable over the Brouwer patent in view of the teachings of the Haley patent and further in view of the teachings of the Knorr reference.

# VII. GROUPING OF CLAIMS

The Claims can be considered in the following groups for purposes of this appeal:

(I) Group I: Claims 1, 5-8, 15, 17, 19, 20, and 22;

(II) Group II: Claims 2, 9-13, 16, 18, 21, and 23-24; and

(III) Group III: Claim 14.

# VIII. ARGUMENT

ARGUMENT (1)

With regard to issue (1), whether the Examiner erred in applying the teachings of the Brouwer patent to the claim language, Appellants will first summarize the teachings of the Brouwer patent, and will then discuss the Examiner's interpretation of those teachings as applied to the language of the claims.

The Brouwer patent teaches a system and method for testing hardware or software applications wherein the test scripts are created by a script writer; the test scripts are

stored at the test case generator; and, the test case generator deploys a test case for an application. As is clearly taught by Brouwer, the test scripts are not created by the test case generator. Rather, the test case generator is "...provided with a plurality of test scripts which, save for a few utility libraries, are created by the user" (see: Col. 2, lines 50-53). Brouwer further teaches that its system "...allows the user...to...develop test scenarios" (Col. 3, lines 5-10) and that "script writing...[is to] be simple enough for one of ordinary skill" (see: Col. 3, lines 36-37). Brouwer then further details the test writer's interface (Col. 4, lines 1-3); utilities accessible to the script writer (Col. 4, lines 18-19; Col. 5, lines 64; Col. 6, lines 61-62; Col. 8, lines 51-54 and line 64; Col. 9, lines 6-7); user created and retrievable semaphore handles (Col. 4, lines 20-21); user defined dispersion patterns (Col. 4, lines 25-26); user's full control of the resources (Col. 4, line 55); user or test engineer design of the test (Col. 6, lines 15-17); automatic event generation based on user defined processes (Col. 16, lines 31-34); user allocation of devices (Col. 18, lines 29-31, Col. 19, lines 40-41, and Col. 20, lines 16-18); and, in the process flow discussed in Col. 28, user selection of actions from a main

menu, user input to information screens, and user modifications to the tests (Col. 28, lines 46-67). The Brouwer description further states that "the [system]...gives the user as much customization capability as possible" (Col. 7, lines 42-44). All of the foregoing passages clearly show that, under the Brouwer system and method, test scripts are created by users/script writers.

The Brouwer system includes a "test case generator", however, the role of the Brouwer test case generator is not to generate or create test scripts. Rather, the Brouwer test case generator stores user-created test scripts and "generates" a test case for an application by deploying a test script which has already been created and stored. As expressly taught by Brouwer, the test case generator is "...provided with a plurality of test scripts which, save for a few utility libraries, are created by the user" (see: Col. 2, lines 50-53).

Moreover, the Brouwer test case generator does not perform the function of allocating test scripts to devices. Rather, as expressly taught by Brouwer at Col. 3, lines 54, "[e]ach script manipulates at least one device" and "[each] script...is capable of retrieving access to that device...[and] allocates their own devices" (Col. 4, lines

42-48). While Brouwer teaches that scripts may allocate their own devices, user modification of device allocation is also taught in Brouwer at Col. 15, lines 13-15, wherein it states that a "user must be able to determine that a legitimate reason exists for starting the process on a remote machine", at Col. 18, lines 29-31 wherein it is taught that the test engineer queries about device, as well as at Col. 19, lines 40-41 and Col. 20, lines 16-18 wherein it is stated that the "user allocates devices".

Appellants respectfully assert that the Brouwer patent teaches a system wherein test scripts are written by users or script writers. Under Brouwer, test scripts are not generated by the test case generator. In addition, the Brouwer patent clearly teaches that test scripts are stored at the test case generator, but are neither created nor modified by the test case generator. Further, Brouwer clearly teaches that scripts are allocated to devices either by the scripts themselves or by a user. Brouwer also expressly teaches that "the user is allowed full control of the resources through the resource allocation utility" (Col. 4, line 55 et seq) and that the utility can generate an "unavailable resource error", which could not possibly occur in a system wherein a test script was dynamically created

based on dynamically-determined knowledge of all currently available resources.

Applicants respectfully contend that the Examiner erred in applying the teachings of the Brouwer patent to the claim language. While Claims 1, 2, 5-13 and 15-24 have been rejected as unpatentable over Brouwer in view of Haley, it is the Brouwer patent which has been used to reject most of the claim features. Accordingly, the applicability of Brouwer teachings to the claim language will first be discussed, with application of Haley to the claimed resource process to be discussed below in argument (2).

In response to the Examiner's citations of Brouwer patent teachings as applied to the language of the independent claims, Applicants note the following:

(i) The Examiner asserted that a test bucket for storing sets of test data is supposedly taught at Col. 1, lines 53-67. However, a review of those teachings shows no discussion of a test bucket having test data which will be used to generate test execution scripts. The cited passage summarizes the Brouwer system as including a test cast generator, a multi-layered interface, means for executing test cases, and means for recording test data. No details are provided about

the enumerated components, and no mention is made of a test bucket. Rather, in a subsequent passage, Brouwer teaches that its test case generator stores test scripts which have been created by a user/script writer (Col. 2, lines 63-65).

(ii) With regard to the claimed job receiver process, for accepting test requests from a user, each test request comprising an identifier for selecting test data from the test bucket, Applicants note that the Examiner has cited the same teachings found in Col. 1 at lines 53-67. Those teachings do not mention receiving a test request from a user, let alone receiving a test request having an identifier for selecting test data from a test bucket. Rather, since the Brouwer user creates the test script, no test request identifier would be applicable. A Brouwer user is creating the test script, not requesting that a job execution process automatically create the test script.

(iii) With regard to the claimed resource process and resource pool for managing system resource data to indicate resources available for software testing on a set of client computer systems, the Examiner has cited Col. 4, lines 43-62. Applicants note that the cited

passage expressly teaches that a Brouwer test script retrieves access to a device. Clearly such is not the same as or suggestive of a test script which is dynamically created based on test data and available devices. The Brouwer test script could not allocate its own device if the test script was not created until available devices were known.

iv) With regard to the job execution process for creating test execution script data based on the test data identified in the test request and on the available resources, wherein the job execution process receives the test request from the job receiver process, dynamically creates the test execution script based upon the resource pool indicating the availability of resources required for the execution of the test on one or more of the set of client computer systems, and initiates testing by forwarding the test execution script data to the appropriate one or more of the set of client computer systems, Appellants note that the cited teachings from Col. 3, lines 53-67 discuss synchronizing between test scripts. The cited passage does not teach or suggest automatically creating test execution script data based on test data

and available resources. Moreover, if Brouwer's scripts allocate their devices, and subsequent synchronization is needed, then clearly the scripts have not been created dynamically with knowledge of the available resources.

Appellants believe that the Examiner has erred in applying the teachings of the Brouwer patent to the claim language. While Brouwer uses some of the same vocabulary as the present claims, Brouwer's teachings clearly are not the same as nor suggestive of that which is being claimed. The Examiner has extracted isolated vocabulary from Brouwer and attempted to apply those terms for rejecting the claim language. However, it is clear that the isolated terms are not being correctly applied to the claim language. Brouwer is describing a system for a user to create test scripts which will be saved at the "test case generator". The Brouwer test scripts are allocated to resources by the scripts, themselves, or by the users. Clearly Brouwer is not teaching or suggesting a system, method, and program storage device wherein a job execution process automatically creates test execution script data based on the test data identified in a test request and based on the resources which have been dynamically determined to be available.

ARGUMENT (2)

Appellants next assert that the Examiner erred in concluding that Claims 1-2, 5-13, and 15-24 are unpatentable over the Brouwer patent in view of the teachings of the Haley patent. Appellants rely on the arguments presented above with regard to the Examiner's interpretation of the teachings of the Brouwer patent and with regard to the erroneous application of those teachings to the claim features. Applicants reiterate that the Brouwer patent does not teach or suggest a method, program storage device, or system for automated testing of software as claimed. Brouwer does not teach the system server component or functionality of a test bucket for storing sets of test data, a job receiver process for accepting test requests from a user, each test request comprising an identifier for selecting test data from the test bucket, a resource process and resource pool for managing system resource data to indicate resources available for software testing on a set of client computer systems, or a job execution process for creating test execution script data based on the test data identified in a test request and the available resources.

The Examiner has acknowledged that Brouwer does not disclose a process indicating resources available for software testing and has additionally cited the Haley patent against that one claim feature. While the Haley patent does disclose a process for indicating locally available resources, Appellants respectfully assert that the combination of Haley and Brouwer would still not obviate the invention as claimed. Neither Brouwer nor Haley teaches or suggests the means or steps for a test bucket for storing sets of test data, a job receiver process, for accepting test requests from a user, each test request comprising an identifier for selecting test data from the test bucket, a resource process and resource pool for managing system resource data to indicate resources available for software testing on a set of client computer systems, and a job execution process for creating test execution script data based on the test data identified in the test request and on the available resources, as recited in all of the independent claims.

With regard to the claimed resource process and functionality thereof, even if one were to modify Brouwer with the teachings of the Haley patent, the combination would not result in the invention as claimed. If Brouwer

were modified with the Haley resource monitoring, the Haley availability information about local resources would be provided to the Brouwer user during test script writing, consistent with the Brouwer patent teachings regarding test engineer queries about devices found at Col. 18, lines 29-31, and the Brouwer patent teachings regarding user allocation of devices found at Col. 19, lines 40-41 and Col. 20, lines 16-18. The resource availability information would not, however, be provided for use in dynamically and automatically generating test scripts by a job execution process, as is expressly recited in all of the pending claims. In addition, since neither Brouwer nor Haley provides a client-server system, identifying resources on a client would simply not be a logical modification. Clearly, the combination would not yield the invention as set forth in the pending independent claims.

Appellants further assert that, in accordance with well established precedent under the U. S. Patent Law, references which do not obviate the language of the independent claims cannot be said to obviate the language of claims which depend therefrom and add further limitations thereto. Accordingly, Appellants respectfully contend that independent Claims 1, 15, 17 and 19, as well as dependent

Claims 2, 5-13, 16, 18, and 20-24 are allowable over the cited art.

Appellants have grouped Claims 1, 2, 5-13, and 15-24 into two different groups for purposes of the appeal. Group I includes Claims 1, 5-8, 15, 17, 19-20, and 22 which recite the system, program product, and method at the server side. Group II includes Claims 2, 9-13, 16, 18, 21, and 23-24 which additionally recite client process means and steps. Appellants respectfully assert that the combination of Brouwer and Haley does not teach or suggest the claimed client process means and functionality. Neither patent is directed to a client-server architecture. Nonetheless, the Examiner has generally rejected the client-related claim features by citing the Brouwer "client configuration file pre-testing an post-testing (*sic*)" citing Fig. 1, units 28 and 26. Appellants have reviewed the figure, the reference numerals, and the Brouwer text, but cannot find any client process teachings. As illustrated in Fig. 1, the Brouwer system is a testing system at a single location. Brouwer does not teach or suggest that its testing system have remote components located at a client. Unit 28 is the Log/metric and Configuration Database and unit 26 is the Logging and Metrics/Alarm Utility. The units are parts of

the single Brouwer system. While a database may be stored on a remote or a local storage device, there is nothing in Brouwer to suggest that the database 28 is remotely located, let alone that it is on a client computer. Moreover, even if the units 26 and 28 were remotely located, Brouwer does not teach or suggest that those units are client processes with the functionality as claimed. Appellants simply cannot understand how the cited Brouwer units are being applied to the claimed client process language. Appellants conclude, therefore, that the claim features are not obviated by the cited units.

In response to all of the arguments which Applicants had submitted to overcome the rejections from the most recent Office Action, the Examiner's sole response is to state that "Brouwer discloses `test request and the available resources'" at Col. 4, lines 41-67. The cited passage discusses resource allocation by the scripts and the users. The cited passage does not teach or suggest that a test request with identifier and current resource availability information be used to automatically and dynamically create test scripts. Appellants fail to understand how the cited passage supports the Examiner's obviousness rejection.

In light of the foregoing arguments, Appellants request that the rejections of Claims 1-2, 5-13, and 15-24 based on the combination of Brouwer and Haley be overruled.

ARGUMENT (3)

Finally, Appellants assert that the Examiner erred in concluding that Claim 14 is unpatentable over the Brouwer patent in view of the teachings of the Haley patent and further in view of the teachings of the Knorr reference. With regard to the rejection of Claim 14, Appellants rely on the arguments presented above with respect to the combination of teachings from Brouwer and Haley. Moreover, the Knorr reference does not provide the teachings which are missing from the combination of Brouwer and Haley. The Knorr reference discloses the use of a DOS system with an essential set of programs to enable a computer to run. Those teachings would not, however, motivate one skilled in the art to modify the Brouwer and Haley combination in such as way as to render Claim 14 obvious. Claim 14 recites the client process of Claim 2, additionally comprising a control process (Claim 12), an automated machined refresh subsystem responsive to a refresh command (Claim 13), wherein the automated machine refresh subsystem is for DOS-based client

system computers running a non-DOS operating system for software testing, the automated machine refresh subsystem comprising a stored machine image, a refresh script for modification of the boot.ini and autoexec.bat files on client system computers, and wherein the modified autoexec.bat file is configured to modify the boot.ini file and execute drive image software for loading the stored machine image and for rebooting the system into the non-DOS operating system for software testing. One having skill in the art would not be motivated to apply the Knorr DOS teachings to modify Brouwer, or Brouwer as modified by Haley, since neither Brouwer nor Haley teaches nor suggests a client location or client process. If one were motivated to apply the Knorr teachings to Brouwer (or Brouwer as modified by Haley), one might modify the Brouwer testing system to use the Knorr instructions to re-boot the Brouwer test system from a DOS to a non-DOS operating system. However, such a modification would not result in the invention as set forth in Claim 14. Accordingly, Applicants believe that Claim 14 is not rendered obvious by the combined teachings of the cited references.

CONCLUSION


Applicants respectfully assert that the Examiner has

erred in interpreting the teachings of the Brouwer patent,

has erred in applying the teachings of the Brouwer patent to

the language of the claims, has erred in concluding that the

combined teachings of Brouwer and Haley obviate Claims 1, 2,

5-13, and 15-24, and has erred in concluding that the

combination of Brouwer and Haley would logically be modified

by Knorr, or that the combination would obviate Claims 14.

In light of the foregoing arguments, Appellants request that

the decision of the Examiner, rejecting all of the pending

claims, be overturned by the Board and that the claims be

passed to issuance.


Respectfully submitted,
C. Conan, et al

By:  Anne Vachon Dougherty
Registration No. 30,374
Tel. (914) 962-5910

APPENDIX OF CLAIMS

1.    A system for automated testing of software, the
system comprising a system server component comprising,

a test bucket for storing sets of test data,

a job receiver process, for accepting test requests
from a user, each test request comprising an identifier for
selecting test data from the test bucket,

a resource process and resource pool for managing
system resource data to indicate resources available for
software testing on a set of client computer systems,

a job execution process for creating test execution
script data based on the test data identified in a test
request and the available resources,

wherein the job execution process receives the test
request from the job receiver process and receives input
from the resource process indicating resources available for
software testing,

dynamically creates the test execution script based
upon the resource pool indicating the availability of
resources required for the execution of the test on one or
more of the set of client computer systems, and

initiates testing by forwarding the test execution script data to the appropriate one or more of the set of client computer systems, and

the system server component further comprising a means for accepting and storing test results from the set of client computer systems.

2.      The system of claim 1 further comprising a client process component, the client process component being executable on one or more of the set of client computer systems and comprising

a listener process for accepting test execution script data from the system server component,

a test execution process for carrying out the testing specified by test execution script data provided by the listener process, and for generating a test report and for communicating the test report to the system server component.

5.      The system of claim 1, in which the system server further comprises an active job queue and a dispatcher process,

job receiver process placing test requests on the active job queue upon receipt,

the dispatcher process determining when a subject test request on the active job queue is matched by available system resources as indicated by the resource pool and providing the subject test request to the job execution process.

6.    The system of claim 5 further comprising a complete job queue for receiving test requests from the job execution process upon the completion of the testing defined by the test request.

7.    The system of claim 1 in which the system server component further comprises a database for the storage of test results received by the job execution process.

8.    The system of claim 1 in which the system server comprises TCP/IP sockets for accepting test requests and communicating with the set of client systems.

9.    The system of claim 2 in which the listener process generates a test script file from the test script data

received from the system server component and which test script file used by the test execution process to define the testing carried out by the client process.

10. The system of claim 2 in which the test execution process generates a report file which is returned to the system server on completion of testing.

11. The system of claim 2 in which the client process further comprises a client configuration file for selectively defined pre-testing and post-testing configuration of the client system computer on which the client process is executing.

12. The system of claim 2 in which the client process further comprises a control process for receiving control queries and commands from the system server component and for responding to the control queries and commands and in which the job execution process in the system server component further comprises means for generating control queries and commands and for receiving responses to the control queries and commands.

13. The system of claim 12 in which the control queries and commands comprise a refresh command, the system further comprising an automated machine refresh subsystem responsive to the refresh command.

14. The system of claim 13 in which the automated machine refresh subsystem is for DOS- based client system computers running a non-DOS operating system for software testing, the automated machine refresh subsystem comprising

a stored machine image,

a refresh script for modification of the boot.ini and autoexec.bat files on client system computers,

the modified autoexec.bat file being configured to modify the boot.ini file and execute drive image software for loading the stored machine image and for rebooting the system into the non-DOS operating system for software testing.

15. A computer program product for use with a computer comprising a central processing unit and random access memory, said computer program product comprising a computer usable medium having computer readable code means embodied

in said medium for software testing in distributed systems,
said computer program product comprising:

computer readable program code means for causing a
computer to define and manage a test bucket for storing sets
of test data,

computer readable program code means for causing a
computer to execute a job receiver process, for accepting
test requests from a user, each test request comprising an
identifier for selecting test data from the test bucket,

computer readable program code means for causing a
computer to execute a resource process for managing system a
resource pool to indicate resources available for software
testing on a set of client computer systems,

computer readable program code means for causing a
computer to execute a job execution process for creating
test execution script data based on the test data identified
in a test request and the available resources,

wherein the job execution process receives the test
request from the job receiver process and an indication of
the available resources from the resource process,

dynamically creates the test execution script data
based the resource pool indicating the availability of

resources required for the execution of the test on one or more of the set of client computer systems, and

initiates testing by forwarding the test execution script data to the appropriate one or more of the set of client computer systems, and

computer readable program code means for causing a computer to accept and store test results from the set of client computer systems.

16.    The computer program product of claim 15 further comprising

computer readable program code means for causing a computer to execute a client process component, the client process component being executable on one or more of the set of client computer systems and comprising

computer readable program code means for causing a computer to execute a listener process for accepting test execution script data from the system server component,

computer readable program code means for causing a computer to execute a test execution process for carrying out the testing specified by test execution script data provided by the listener process, and for generating a test

report and for communicating the test report to the system server component.

17.    A computer program product tangibly embodying a program of instructions executable by a computer for implementing a system for automated testing of software, the system comprising a system server component comprising,

a test bucket for storing sets of test data,

a job receiver process, for accepting test requests from a user, each test request comprising an identifier for selecting test data from the test bucket,

a resource process and resource pool for managing system resource data to indicate resources available for software testing on a set of client computer systems,

a job execution process for creating test execution script data based on the test data identified in a test request and the available resources by the steps of:

receiving the test request from the job receiver process and an indication of available resources from the resource process,

dynamically creating test execution script data based upon the resource pool indicating the availability of

resources required for the execution of the test on one or more of the set of client computer systems, and

initiating testing by forwarding the test execution script data to the appropriate one or more of the set of client computer systems, and

the system server component further comprising a means for accepting and storing test results from the set of client computer systems.


18. The computer program product of claim 17, the system for automated testing of software further comprising a client process component, the client process component being executable on one or more of the set of client computer systems and comprising

a listener process for accepting test execution script data from the system server component,

a test execution process for carrying out the testing specified by test execution script data provided by the listener process, and for generating a test report and for communicating the test report to the system server component.

19. A method for use with a computer comprising a central processing unit and random access memory, said computer program product comprising a computer usable medium having computer readable code means embodied in said medium for software testing in distributed systems, said method comprising the steps at said computer of:

defining and managing a test bucket for storing sets of test data,

executing a job receiver process, for accepting test requests from a user, each test request comprising an identifier for selecting test data from the test bucket,

executing a resource process for managing a system resource pool to indicate resources available for software testing on a set of client computer systems,

executing a job execution process for creating test execution script data based on the test data identified in a test request and the available resources, by performing the steps of:

receiving the test request from the job receiver process at the job execution process and resource availability from the resource process,

dynamically creating a test execution script indicating the availability of resources required for the execution of

the test on one or more of the set of client computer systems, and

initiating testing at said job execution process by forwarding the test execution script data to the appropriate one or more of the set of client computer systems, and

accepting and storing test results from the set of client computer systems.


20. The system of claim 1 further comprising a web servlet component providing a graphical user interface for use by the user in defining a test request.


21. The system of claim 2 further comprising a web servlet component providing a graphical user interface for use by the user in defining a test request.


22. The system of claim 1 further comprising a parser component for parsing ASCII format test requests defined by the user.


23. The system of claim 2 further comprising a parser component for parsing ASCII format test requests defined by the user.

24. The system of claim 3 further comprising a parser component for parsing ASCII format test requests defined by the user.